Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 22, 2013

A. Grange
H. Alvestrand
Google

February 18, 2013

A VP9 Bitstream Overview
draft-grange-vp9-bitstream-00

Abstract

This document describes VP9, a video codec being developed specifically to meet the demand for the consumption of video over the Internet, including professionally and amateur produced video-on-demand and conversational video content. VP9 is an evolution of the VP8 video codec that is described in [bankoski-rfc6386] and includes a number of enhancements and new coding tools that have been added to improve the coding efficiency. The new tools that have been added so far include: larger prediction block sizes up to 64x64, various forms of compound INTER prediction, more modes for INTRA prediction, ⅛-pel motion vectors, 8-tap switchable sub-pixel interpolation filters, improved motion reference generation, improved motion vector coding, improved entropy coding including frame-level entropy adaptation for various symbols, improved loop filtering, the incorporation of the Asymmetric Discrete Sine Transform (ADST), larger 16x16 and 32x32 DCTs, and improved frame level segmentation. VP9 is under active development and this document provides only a snapshot of the current state of the coding tools as they exist today. The finalized version of the VP9 bitstream may differ considerably from the description contained herein and may encompass the exclusion or modification of existing coding tools or the addition of new coding tools.

Status of this Memo

This Internet-Draft will expire on August 22, 2013.

Copyright Notice

Table of Contents

1. Introduction

Video data accounts for a significant proportion of all internet traffic, and the trend is toward higher quality, larger format and often professionally produced video, encoded at higher data rates and supported by the improved provisioning of high bandwidth internet connections. VP9 is being developed as an open source solution tailored to the specific characteristics of the internet, under the auspices of the WebM project [Google-webm], with the aim of providing the highest quality user experience and the ability to support the widest range of use-cases on a diverse set of target devices. This document provides a high-level technical overview of the coding tools that will likely be included in the final VP9 bitstream.

2. Outline of the Codec

A large proportion of the advance that VP9 has made over VP8 can be attributed to a straightforward generational progression from the current to the future, driven by the need for the greater efficiency required to handle a new coding "sweet-spot" that has evolved to support the provisioning of larger frame size, higher quality video formats.

2.1. Prediction Block Size

A large part of the coding efficiency improvements achieved by VP9 can be attributed to the introduction of larger prediction block sizes. Specifically, VP9 introduces the notion of Super-Blocks of

size up to 64x64 and their quad-tree like decomposition all the way down to a block size of 4x4, with some quirks as described below. In particular, a superblock of size 64x64 (SB64) could be split into 4 superblocks of size 32x32 (SB32), each of which can be further split into 16x16 macroblocks (MB). Each SB64, SB32 or MB could be predicted as a whole using a conveyed INTRA prediction mode, or an INTER prediction mode with up to two motion vectors and corresponding reference frames, as described in Section 3.2. A macroblock can be further split using one of three mode families: B_PRED - where each 4x4 sub-block within the MB can be coded using a signaled 4x4 INTRA prediction mode; I8X8_PRED - where each 8x8 block within the MB can be coded using a signaled 8x8 INTRA prediction mode; and SPLITMV - where each 4x4 sub-block within the MB is coded in INTER mode with a corresponding motion vector, but with the option of grouping common motion vectors over 16x8, 8x16, or 8x8 partitions within the MB. Note that the B_PRED and SPLITMV modes in VP9 work in the same way as they do in VP8.

2.2. Prediction Modes

VP9 supports the following prediction modes for various block-sizes:

2.2.1. INTRA modes

At block-size 4x4, VP9 supports ten intra prediction modes; DC, Vertical, Horizontal, TM (True Motion), Horizontal Up, Left Diagonal, Vertical Right, Vertical Left, Right Diagonal, and Horizontal Down (the same set defined by VP8). For blocks from 8x8 to 64x64 there is also support for ten intra modes; DC, Vertical, Horizontal, TM (True Motion), and six angular predictors corresponding, approximately, to angles of 27, 45, 63, 117, 135, and 153 degrees. Furthermore, there is additionally the option of applying a low-pass filter to the prediction that can be signaled in the bitstream.

2.2.2. INTER Modes

VP9 currently supports INTER prediction from up to three reference frame buffers (named LAST_FRAME, GOLDEN_FRAME and ALTREF_FRAME, as in VP8), but for any particular frame the three available references are dynamically selectable from a pool of eight stored reference frames.

A syntax element in the frame header indicates which sub-set of three reference buffers are available when encoding the frame. A further syntax element indicates which of three frame buffers, if any, are to be updated at the end of encoding a frame. Some coded frames may be designated as invisible in the sense that they are only used as a reference and never actually displayed, akin to the ALTREF frame in VP8. It is also likely that the number of available working reference buffers will be increased from three to four in the final VP9 bitstream.

Each INTER coded block within a frame, may be coded using up to two motion vectors with two different reference buffers out of the three working reference buffers selected for the frame. When a single motion vector is used, sub-pixel interpolation from the indicated reference frame buffer is used to obtain the predictor. When two motion vectors, mv1 and mv2, are conveyed for any given block, the corresponding reference frame buffers ref1 and ref2 must be different from each other, and the final predictor is then obtained by averaging the individual predictors from each of the motion vectors, i.e.,

P[i, j] = floor((Pmv1, ref1[i, j] + Pmv2, ref2[i, j] + 1) / 2)

where P[i, j] is the predictor value at pixel location [i, j], and Pmv1, ref1 and Pmv2, ref2 are the INTER predictors corresponding to the two motion vectors and reference buffers conveyed.

2.2.3. Compound INTER-INTRA Mode

A further prediction mode under consideration is a combination INTER/INTRA mode. In this mode, an INTER predictor and an INTRA predictor are combined in a manner whereby pixels closer to the INTRA prediction edge (top or left) are weighted more heavily towards the INTRA predictor, whilst pixels further away from the edges are weighted more heavily towards the INTER predictor. The exact weights used for each pixel thus depend on the particular INTRA prediction direction in use. Conceptually, each INTRA prediction mode at a given block size is associated with a constant weighting block of the same size - that provides the weight for the INTRA predictor as compared to the inter predictor. For instance, if the weighting matrix for a given INTRA mode m and block-size n is given by an nxn matrix, Wm, with values between 0 and 1, then the predictor of pixel

[i, j] denoted P[i, j] is obtained by:

P[i, j] = Wm[i, j] . Pm[i, j] + (1 - Wm[i, j]) . Pmv, ref[i, j]

where Pm is the INTRA predictor for the given INTRA mode, and Pmv, ref is the INTER predictor obtained using motion vector mv and reference frame index ref. This mode is restricted to one motion vector per block, and only to blocks of size 16x16 and above, i.e. MB/SB32/SB64. The weighting matrix may be obtained from a 1-D exponential decay function of the form A + B exp (-Kx), where x represents the distance along the prediction direction to the nearest left/top edge.

2.3. Sub-Pixel Interpolation

The filters used for sub-pixel interpolation of fractional motion are critical to the performance of a video codec. The maximum motion vector precision supported is 1/8-pixel, with the option of switching between 1/4-pixel and 1/8-pixel precision using a frame level flag. If 1/8-pixel precision is used in the frame, however, it is only used for small motion, depending on the magnitude of the reference motion vector. For larger motion - indicated by a larger reference - there is almost always motion blur which obviates the need for higher precision interpolation. VP9 defines a family of three 8-tap filters, selectable at either the frame or macroblock level in the bitstream:

- 8-tap Regular: An 8-tap Lagrangian interpolation filter designed using the int_filt function in MATLAB,

- 8-tap Sharp: A DCT-based interpolation filter with a sharper response, used mostly around sharper edges,

- 8-tap Smooth (non-interpolating): A smoothing filter designed using the windowed Fourier series approach with a Hamming window. Note that unlike the other two filters, this filter is non-interpolating in the sense that the prediction at integer pixel-aligned locations is a smoothed version of the reference frame pixels.

2.4. Transforms

VP9 supports the Discrete Cosine Transform (DCTs) at sizes 4x4, 8x8, 16x16 and 32x32 and removes the second-order transform that was employed in VP8. Only transform sizes equal to, or smaller than, the prediction block size may be specified. Modes B_PRED and 4x4 SPLITMV are thus restricted to using only the 4x4 transform; modes I8X8_PRED and non-4x4 SPLITMV can use either the 4x4 or 8x8 transform; full-size (16x16) macroblock predictors can be coupled with either the 4x4, 8x8 or 16x16 transforms, and superblocks can use any transform size up to 32x32. Further restrictions on the available sub-set of transforms can be signaled at the frame-level, by specifying a maximum allowable transform size, or at the macroblock level by explicitly signaling which of the available transform sizes is used.

In addition, VP9 introduces support for a new transform type, the Asymmetric Discrete Sine Transform (ADST), which can be used in combination with specific intra-prediction modes. It has been shown in [Han-Icassp] and [Han-Itip] that when a one-sided boundary is available, as in most INTRA prediction modes, the ADST rather than the DCT is the optimal transform for the residual signal. Intra prediction modes that predict from a left edge can use the 1-D ADST in the horizontal direction, combined with a 1-D DCT in the vertical direction. Similarly, the residual signal resulting from intra prediction modes that predict from the top edge can employ a vertical 1-D ADST transform combined with a horizontal 1-D DCT transform. Intra prediction modes that predict from both edges such as the True Motion (TM_PRED) mode and some diagonal intra prediction modes, use the 1-D ADST in both horizontal and vertical directions.

2.5. Motion Vector Reference Selection and Coding

One of the most critical factors in the efficiency of motion vector encoding is the generation of a suitable reference motion vector to be used as a predictor. VP9 creates a sorted list of candidate reference motion vectors that encompasses the three vectors best, nearest and near as defined by VP8. In addition to the candidates produced by the VP8 algorithm, VP9 additionally evaluates the motion vector of the co-located block in the reference frame and those of nearby blocks. VP9 introduces a new scoring mechanism to rank these reference vectors whereby each candidate is evaluated to determine

how well it would have predicted the reconstructed pixels in close

proximity to the current block (more specifically a small number of rows immediately above the current block, and maybe a small number of columns to the left of the current block). A predictor is created using each candidate vector in turn to displace the pixels in the reference frame and the variance of the resulting error signal, with respect to the set of pixels in the current frame, is used to rank the reference vectors.

With the three best candidate reference vectors best, nearest and near identified, the encoder can either signal the use of the vector identified as the nearest (NEAREST_MV mode) or near (NEAR_MV mode) or, if neither of them is deemed appropriate, signal the use of a completely new motion vector (NEW_MV mode) that is then specified as a delta from the best reference candidate.

One further mode, ZERO_MV, signals the use of the (0, 0) motion vector.

In addition, a more efficient motion vector offset encoding mechanism has been introduced.

2.6. Entropy Coding and Adaptation

The VP9 bitstream employs the VP8 BoolCoder as the underlying arithmetic encoder. Generally speaking, given a symbol from any n-ary alphabet, a static binary tree is constructed with n-1 internal nodes, and a binary arithmetic encoder is run at each such node as the tree is traversed to encode a particular symbol. The probabilities at each node use 8-bit precision. The set of n-1 probabilities for coding the symbol is referred to as the entropy coding context of the symbol. Almost all of the coding elements conveyed in a bit-stream - including modes, motion vectors, reference frames, and prediction residuals for each transform type and size - use this strategy.

Video content is inherently highly non-stationary in nature and a critical component of any codec is the mechanism used to track the statistics of the various encoded symbols and update the parameters of the entropy coding contexts to match. VP9 makes use of forward context updates through the use of flags in the frame header that signal modifications of the coding contexts at the start of each frame. The syntax for forward updates is designed to allow an arbitrary sub-set of the node probabilities to be updated whilst leaving the others unchanged. The advantage of using forward adaptation is that decoding performance can be substantially improved, because no intermediate computations based on encountered token counts is necessary. Updates are encoded differentially, to

allow a more efficient specification of updated coding contexts which is essential given the expanded set of tokens available in VP9.

In addition, there is also a limited option for signaling backward adaptation, which in VP9 is only possible at the end of encoding each frame so that the impact on decoding speed is minimal. Specifically, for every frame encoded, first a forward update modifies the entropy coding contexts for various symbols encoded starting from the initial state at the beginning of the frame. Thereafter, all symbols encoded in the frame are coded using this modified coding state. At the end of the frame, both the encoder and decoder are expected to have accumulated counts for various symbols actually encoded or decoded over the frame. Using these actual distributions, a backward update step is applied to adapt the entropy coding context for use as the baseline for the next frame.

2.7. Loop Filter

VP9 introduces a variety of new prediction block and transform sizes that require additional loop filtering options to handle a larger number of combinations of boundary types. VP9 also incorporates a flatness detector in the loop filter that detects flat regions and varies the filter strength and size accordingly.

2.8. Segmentation

VP9 introduces more advanced segmentation features that make it much more efficient and powerful, allowing each superblock or macroblock to specify a segment-ID to which it belongs. Then, for each segment, the frame header can convey common features that will be applied to all MBs/SB32s/SB64s belonging to the same segment ID. Further, the segmentation map is coded differentially across frames in order to minimize the size of the signaling overhead. Examples of information that can be conveyed for a segment include: restrictions on the reference frames that can be used for each segment, coefficient skips, quantizer and loopfilter strength, and transform size options. Generally speaking, the segmentation mechanism provides a flexible set of tools that can be used, in an application specific way, to target improvements in perceptual quality for a given compression ratio.

In the reference implementation, segmentation is currently used to identify background and foreground areas in encoded video content. The (static) background is then coded at a higher quality compared to the rest of the frame in certain reference frames (such as the alt-ref frame) that provides prediction that persists over a number of

frames. In contrast, for the frames between these persistent reference frames, the background is given fewer bits by, for example,

restricting the set of available reference buffers, using only the ZERO_MV coding mode, or skipping the residual coefficient block. The result is that more bits are available to code the foreground-portion of the scene, while still preserving very good perceptual quality on the static background. Other use cases involving spatial and temporal masking for perceptual quality improvement are conceivable.

3. Bitstream features

In addition to providing high compression efficiency with reasonable complexity, the VP9 bitstream includes features designed to support a variety of specific use-cases that are important to internet video content delivery and consumption. This section provides an overview of these features.

3.1. Error-Resilience

For communication of conversational video with low latency over an unreliable network, it is imperative to support a coding mode where decoding can continue without errors even when arbitrary frames are lost. Specifically, the arithmetic encoder should still be able to decode symbols correctly in frames subsequent to lost frames, even though frame buffers have been corrupted, leading to encoder-decoder mismatch. The hope is that the drift between the encoder and decoder will still be manageable until such time as a key frame is sent or other corrective action (such as reference picture selection) can be taken. VP9 supports a frame level error_resilient_mode flag which when turned on will only allow coding modes where this is possible to achieve. In particular, the following restrictions are imposed in error resilient mode:

1. The entropy coding context probabilities are reset to defaults at the beginning of each frame. (This effectively prevents propagation of forward updates as well as backward updates),

2. For MV reference selection, the co-located MV from a previously encoded reference frame can no longer be included in the reference candidate list,

3. For MV reference selection, sorting of the initial list of motion vector reference candidates based on search in the reference frame buffer is disabled.

These restrictions produce a modest performance drop.

3.2. Parallel Decodability

Smooth encoding and playback of high-definition video on resource constrained personal devices (smartphones, tablets, netbooks, etc.) in software necessitates exploiting some form of parallelism, so that multi-threaded applications can be built around the codec to exploit the inherent multi-processing capabilities of modern processors. This may include either the ability to encode/decode parts of a frame in parallel, or the ability to decode successive frames in parallel, or a combination of both. VP9 supports both forms of parallelism, as described below:

3.2.1. Frame-Level Parallelism

A frame level flag frame_parallel_mode, when turned on, enables an encoding mode where the entropy decoding for successive frames can be conducted in a quasi-parallel manner just by parsing the frame headers, before these frames actually need to be reconstructed. In this mode, only the frame headers need to be decoded sequentially. Beyond that, the entropy decoding for each frame can be conducted in a lagged parallel mode as long as the co-located motion vector information from a previous reference frame has been decoded prior to the current frame. The reconstruction of the frames can then be conducted sequentially in coding order as they are required to be displayed. This mode will enable multi-threaded decoder implementations that results in smoother playback performance. Specifically, this mode imposes the following restrictions on the bitstream, which is a subset of the restrictions for the error-resilient mode.

1. Backward entropy coding context updates are disabled. But forward updates are allowed to propagate.

2. For MV reference selection, sorting of the initial list of motion vector reference candidates based on a search in the reference frame buffer is disabled. However, the co-located MV from a previously encoded reference frame can be included in the initial candidate list.

3.2.2. Tiling

In addition to making provisions for decoding multiple frames in parallel, VP9 also has support for decoding a single frame using multiple threads. For this, VP9 introduces tiles, which are independently coded and decodable sub-units of the video frame. When enabled a frame can be split into, for example, 2 or 4 column-based tiles. Each tile shares the same frame entropy model, but all contexts and pixel values (for intra prediction) that cross tile-

boundaries take the same value as those at the left, top or right edge of the frame. Each tile can thus be decoded and encoded completely independently, which is expected to enable significant speedups in multi-threaded encoders/decoders, without introducing any additional latency. Note that loop-filtering across tile-edges can still be applied, assuming a decoder implementation model where the loop-filtering operation lags the decoder's reconstruction of the individual tiles within the frame so as not to use any pixel that is not already reconstructed. Further, backward entropy adaptation - a light-weight operation - can still be conducted for the whole frame after entropy decoding for all tiles has finished.

3.3. Scalability

The VP9 bit-stream will provide a number of flexible features that can be combined in specific ways to efficiently provide various forms of scalability. VP9 increases the number of available reference frame buffers to eight, from which three may be selected for each frame. In addition, each coded frame may be resampled and coded at a resolution different from the reference buffers, allowing internal spatial resolution changes on-the-fly without having to resort to using keyframes. When such a resolution change is signaled in the bit-stream, the reference buffers as well as the corresponding MV information is suitably transformed to the new resolution before applying standard coding tools. Furthermore, VP9 defines the

maintenance of four different entropy coding contexts to be selected and optionally updated on every frame, thereby making it possible for the encoder to use a different entropy coding context for each scalable layer, if required. These flexible features together enable an encoder/decoder to implement various forms of coarse-grained scalability, including temporal, spatial, or combined spatio-temporal scalability, without explicitly creating spatially scalable encoding modes.

4. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

The VP9 bitstream offers no security functions. Integrity and confidentiality must be ensured by functions outside the bistream.

The VP9 bitstream does not offer functions for embedding of other types of objects, either active or passive. So this class of attack cannot be mounted using VP9.

Implementations of codecs are often written with a strong focus on speed. The reference software has been carefully vetted for security issues, but no guarantees can be given. People who use other people's decoder software will need to take appropriate care when executing the software in a security sensitive context.

6. Acknowledgements

This document is heavily based on the paper by Bankoski, J., Bultje, R.S., Grange, A., Gu, Q., Han, J., Koleszar, J., Mukherjee, D., Wilkins, P., Xu, Y., Towards a Next Generation Open-source Video Codec, IS&T / SPIE EI Conference on Visual Information Processing and Communication IV, February 5-7, 2013.

7. Informative References

[Google-webm]
"WEBM project website", March .

http://www.webmproject.org/

[Han-Icassp]
Han, J., "Towards jointly optimal spatial prediction and adaptive transform in video/image coding", March 2010.

IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP), pp. 726-729

[Han-Itip]
Han, J., "Jointly optimized spatial prediction and block transform for video and image coding", April 2012.

IEEE Transactions on Image Processing, vol. 21, pp. 1874-1884

[RFC6368] Marques, P., Raszuk, R., Patel, K., Kumaki, K., and T. Yamagata, "Internal BGP as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 6368, September 2011.

[vp9-paper]
Bankoski, J., Bultje, R., Grange, A., Gu, Q., Han, J.,

Koleszar, J., Mukherjee, D., Wilkins, P., and Y. Xu, "Towards a Next Generation Open-source Video Codec", February 2013.

IS&T / SPIE EI Conference on Visual Information Processing and Communication IV

Authors' Addresses

Adrian Grange

Google

Email: agrange@google.com

Harald Alvestrand
Google

Phone:
Fax:
Email: hta@google.com
URI: